IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ayoub Bensaid
09/30/2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context:

  SpaceX promotes Falcon 9 rocket launches on its website at a price point of $62 million, significantly undercutting other providers whose costs can exceed $165 million per launch. A substantial portion of this cost advantage arises from SpaceX's ability to recycle the initial stage of the rocket. Consequently, if we can ascertain the likelihood of a successful first-stage landing, we can accurately estimate the overall launch cost. This knowledge becomes invaluable when an alternative company seeks to compete with SpaceX in bidding for rocket launch contracts. The primary objective of this project is to establish a machine learning pipeline designed to predict the probability of a successful first-stage landing

- Problems we want to find answers:

  -What contributes to the successful landing of a rocket?

  -How do different elements interact to impact the landing success rate?

  -What prerequisites must be met for a successful landing program to be ensured?

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

- Information was gathered through the utilization of the SpaceX API and by extracting data from Wikipedia using web scraping techniques.

Perform data wrangling

- Categorical features underwent one-hot encoding.

Perform exploratory data analysis (EDA) using visualization and SQL

Perform interactive visual analytics using Folium and Plotly Dash

Perform predictive analysis using classification models

- How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected:
- - The data collection process began with a requests.get() to the SpaceX API.
- - Following that, we decoded the response content into JSON format using the .json() function call and transformed it into a pandas dataframe using .json_normalize().
- - Subsequently, we conducted data cleaning, identifying and addressing missing values as needed.
- - Additionally, we engaged in web scraping from Wikipedia for Falcon 9 launch records using BeautifulSoup. Our aim was to extract launch records in the form of an HTML table, parse this table, and convert it into a pandas dataframe for subsequent analysis.

Data Collection – SpaceX API

- We employed a GET request to access the SpaceX API for data collection, followed by a process of cleaning the acquired data and performing fundamental data wrangling and formatting tasks.
- the GitHub URL of the completed SpaceX API calls notebook:

https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- We applied web scraping techniques with BeautifulSoup to extract Falcon 9 launch records. Subsequently, we parsed the obtained table and transformed it into a pandas dataframe.

```
# use requests.get() method with the provided static_url
response = requests.get(static_url)
# assign the response to a object

# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, 'html')
```

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_name
tc = first_launch_table.find_all('th')
for th in tc:
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:
            column_names.append(name)
```

the GitHub URL of the completed web scraping notebook:

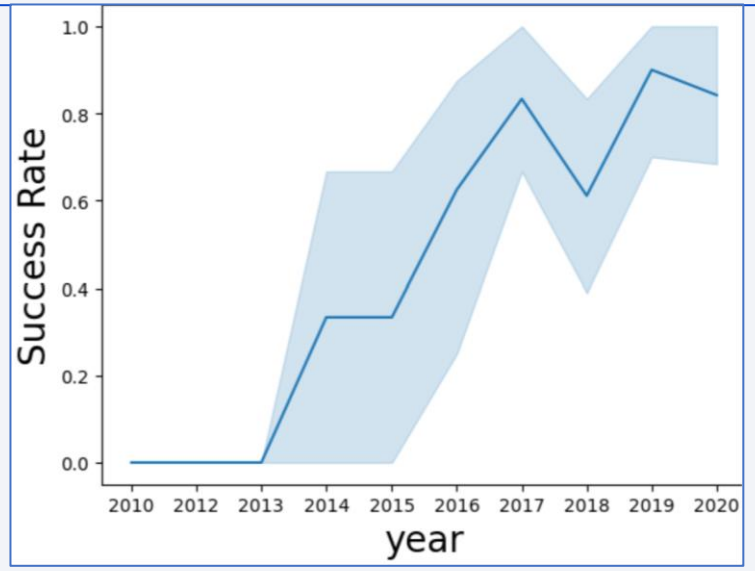
https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/jupyter-labs-webscraping.ipynb

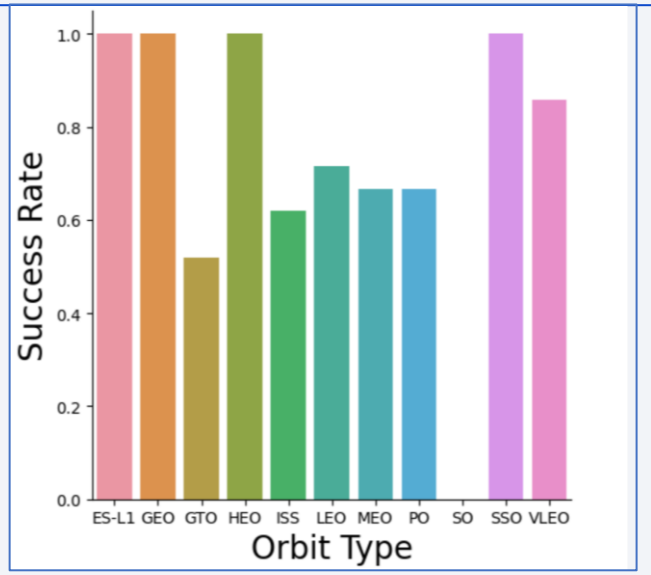
Data Wrangling

- We conducted exploratory data analysis to identify the training labels. This involved calculating the frequency of launches at each site, as well as the occurrence of each orbit type.
- Furthermore, we generated a landing outcome label based on the outcome column and exported the results to a CSV file.
- You can find the notebook by following this link:

https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

EDA with Data Visualization





We delved into the data by creating visualizations to examine various relationships, including those between flight number and launch site, payload and launch site, success rates for each orbit type, flight number and orbit type, and the annual trend in launch success.

the GitHub URL for the completed data visualization notebook:

https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

We seamlessly imported the SpaceX dataset into a PostgreSQL database directly within the Jupyter notebook. Subsequently, we harnessed SQL for exploratory data analysis, using queries to uncover valuable insights from the dataset, including:

- Identifying unique launch site names in the space mission.
- Calculating the total payload mass transported by boosters launched by NASA (CRS).
- Determining the average payload mass carried by booster version F9 v1.1.
- Enumerating the total count of successful and failed mission outcomes.
- Extracting information on failed landing outcomes on drone ships, including their booster versions and launch site names.

You can access the notebook through the following link:
https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We annotated all launch sites and integrated map elements such as markers, circles, and lines on the Folium map to visually represent the outcomes of launches, whether they were successful or unsuccessful.
- We assigned the launch outcomes to distinct classes, designating '0' for failures and '1' for successes. By utilizing color-coded marker clusters, we discerned the launch sites with notably high success rates.
- In addition, we computed the distances between each launch site and its nearby surroundings, allowing us to address specific inquiries, including:
  - Proximity to railways, highways, and coastlines in relation to launch sites.
  - Whether launch sites maintain a certain distance from urban areas.

You can find the notebook by following this link:
https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

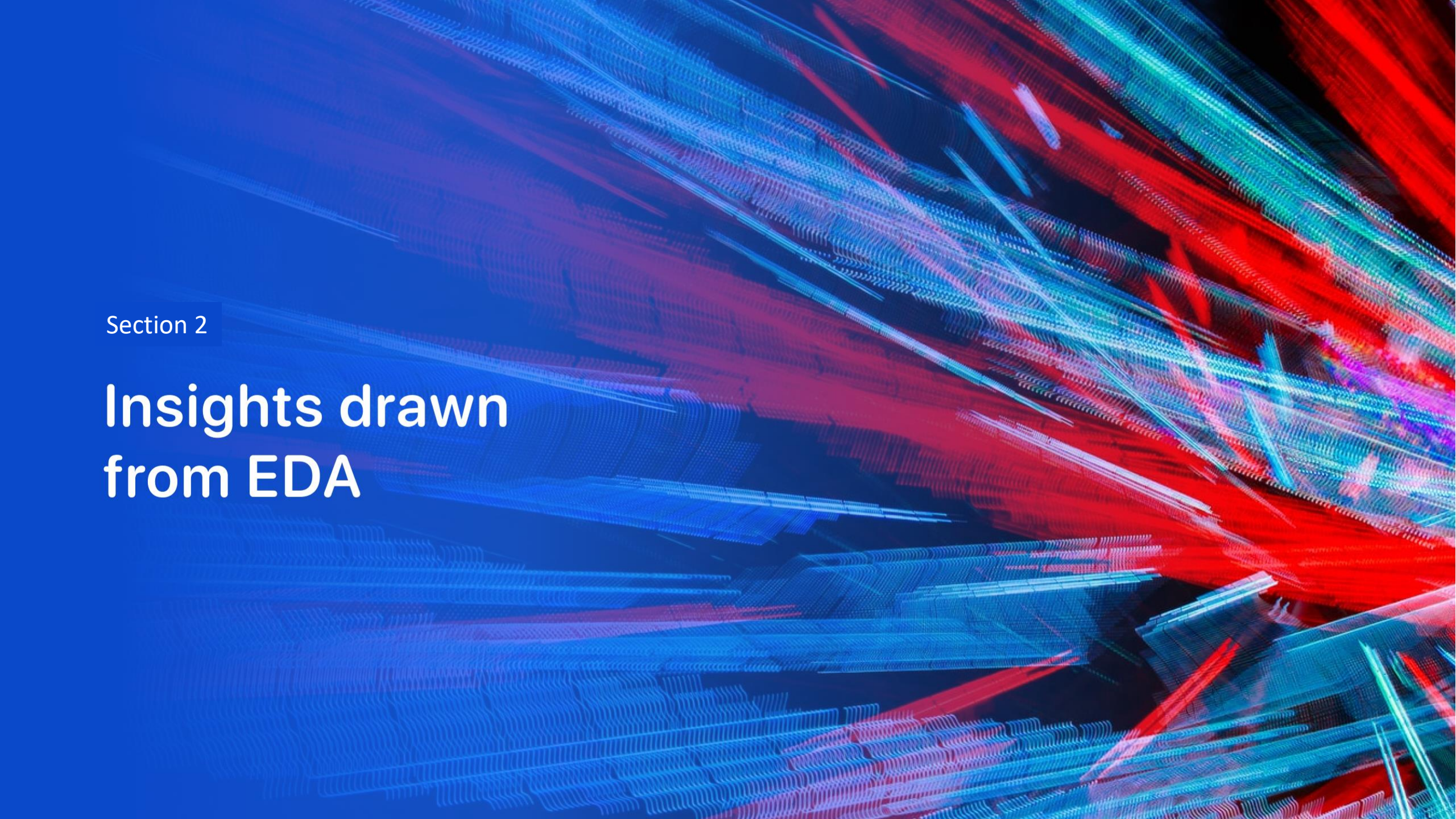
- We developed an interactive dashboard using Plotly Dash, enabling dynamic data exploration.
- Within this dashboard, we generated pie charts that visually depict the total number of launches for specific sites. Additionally, we created scatter graphs that illustrate the relationship between launch outcomes and payload mass (in kilograms) for different booster versions.
- You can access the notebook by following this link: https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- We initiated the data processing by loading and manipulating the dataset with the assistance of numpy and pandas. Subsequently, we partitioned our data into training and testing subsets.
- Following this, we constructed various machine learning models and fine-tuned their hyperparameters through GridSearchCV. Our evaluation criterion for model performance was accuracy. To enhance the model, we engaged in feature engineering and algorithm tuning.
- Ultimately, we identified the classification model that exhibited the highest performance.
- You can access the notebook via the following link: https://github.com/AyoubBenSaid2005/My_submission_assignment/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite%20(1).ipynb

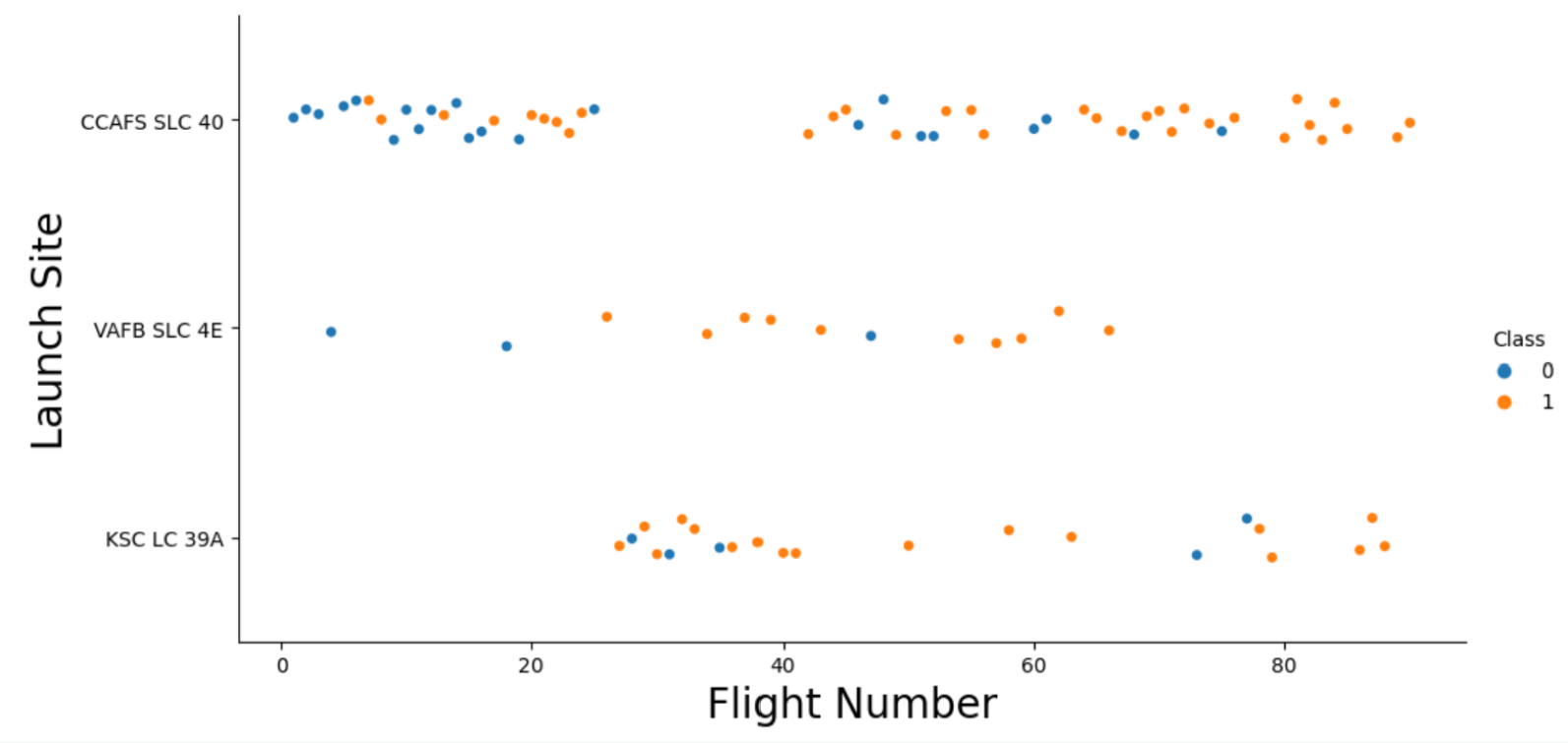
Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

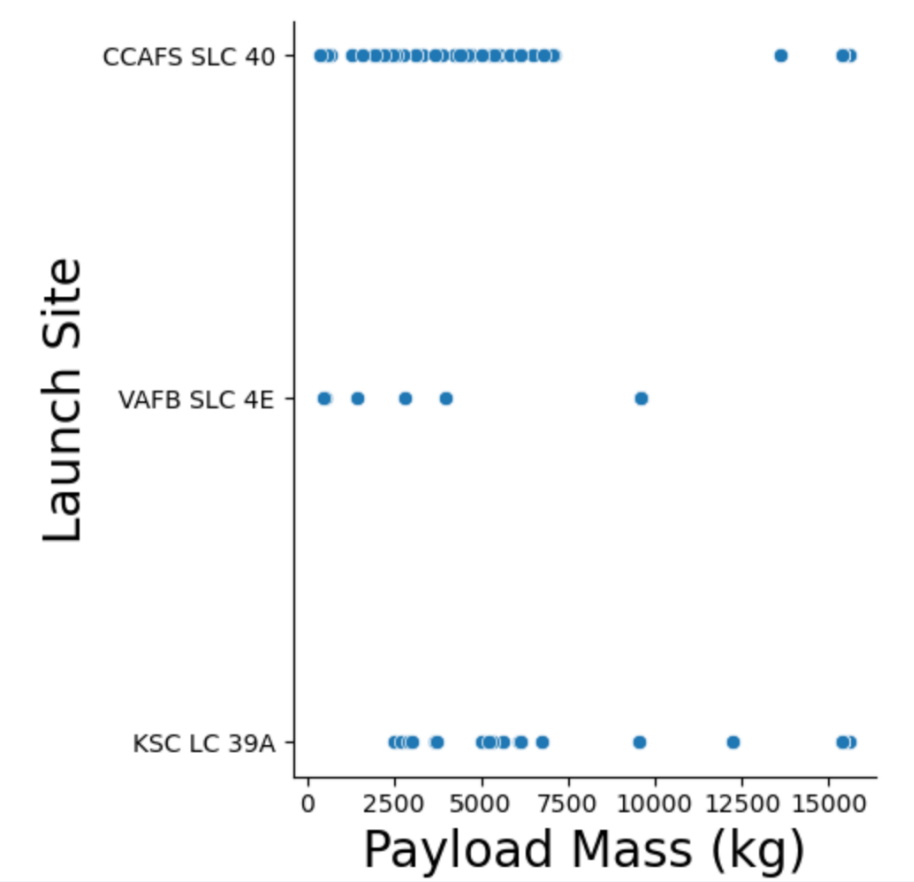
Flight Number vs. Launch Site

The plot revealed a positive correlation: as the number of flights at a launch site increased, so did the success rate at that particular site.



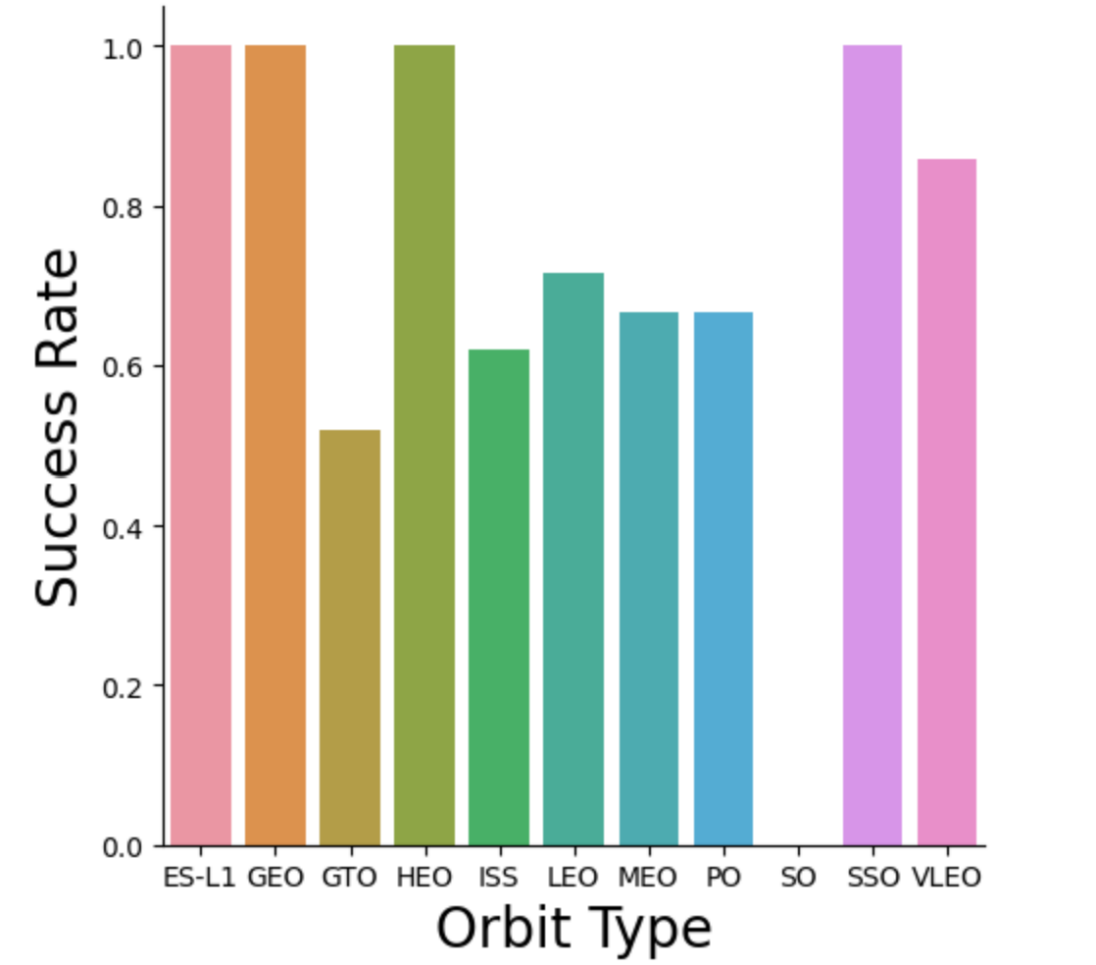
Payload vs. Launch Site

- From the plot, we found that The greater is our Payload Mass , the less is our success rate



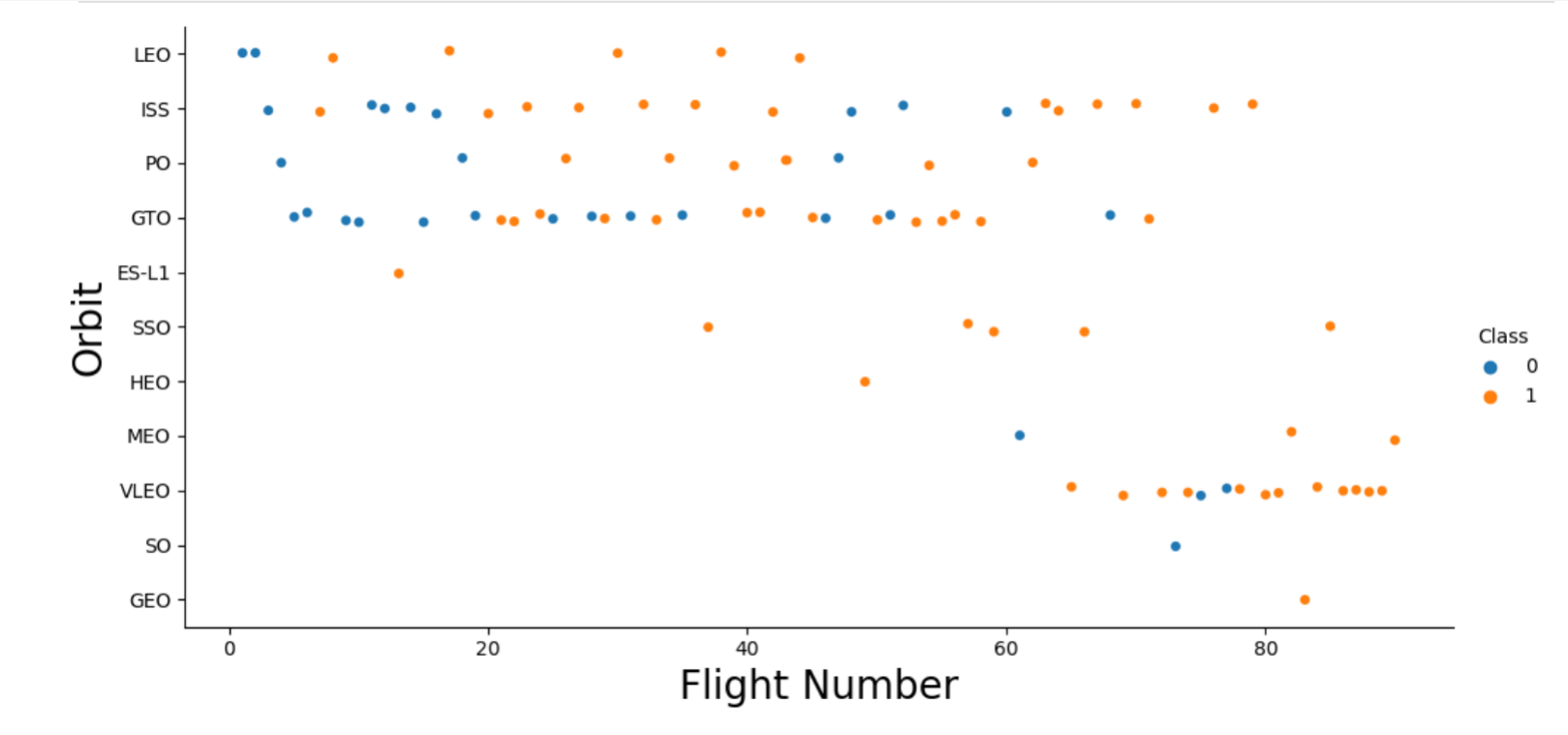
Success Rate vs. Orbit Type

By examining the plot, it becomes evident that ES-L1, GEO, HEO, SSO, and VLEO stood out with the highest success rates.



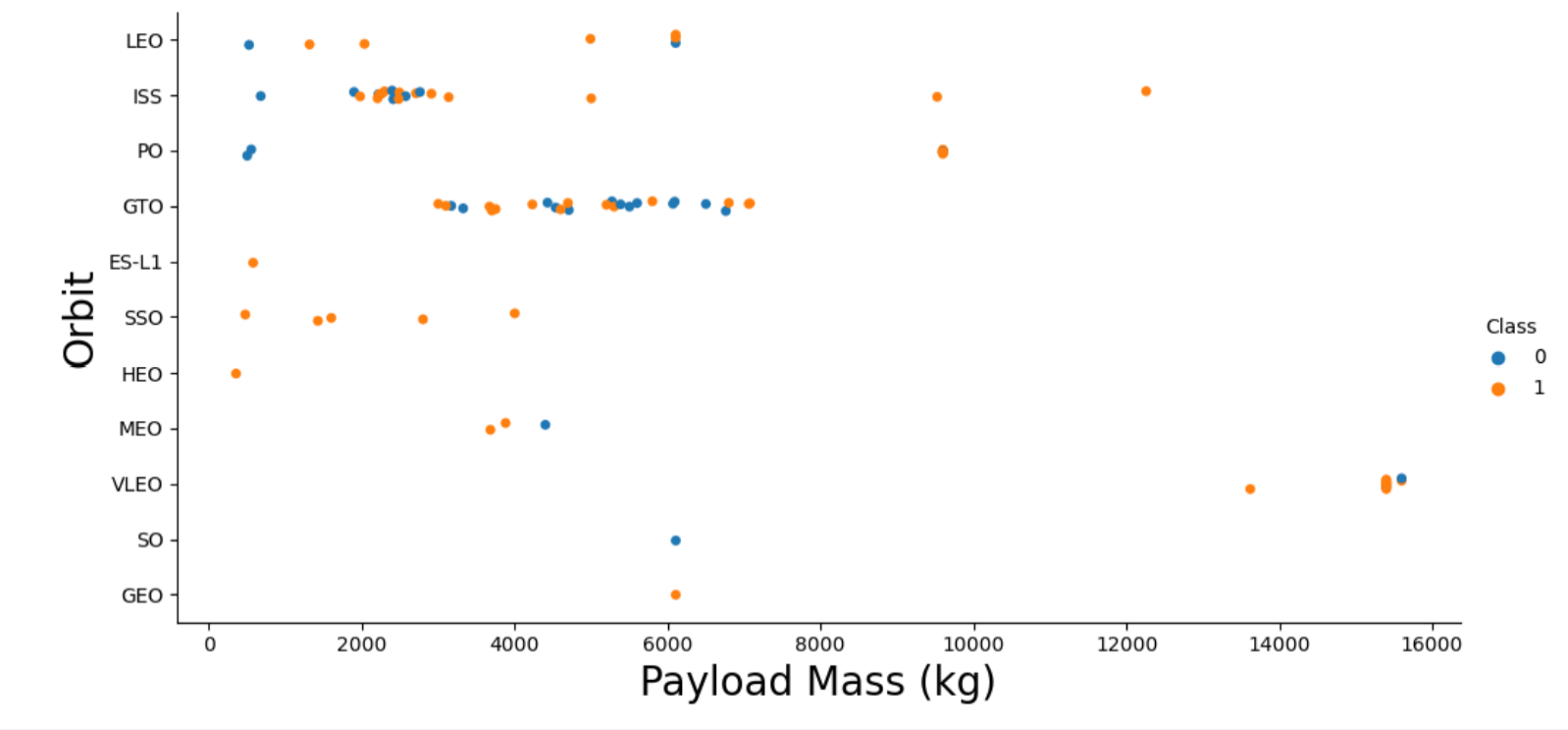
Flight Number vs. Orbit Type

In the presented plot depicting Flight Number versus Orbit type, a noticeable pattern emerges. Specifically, it indicates that within the LEO orbit, the success rate appears to be influenced by the number of flights. Conversely, when considering the GTO orbit, no discernible relationship between flight number and orbit success is evident.



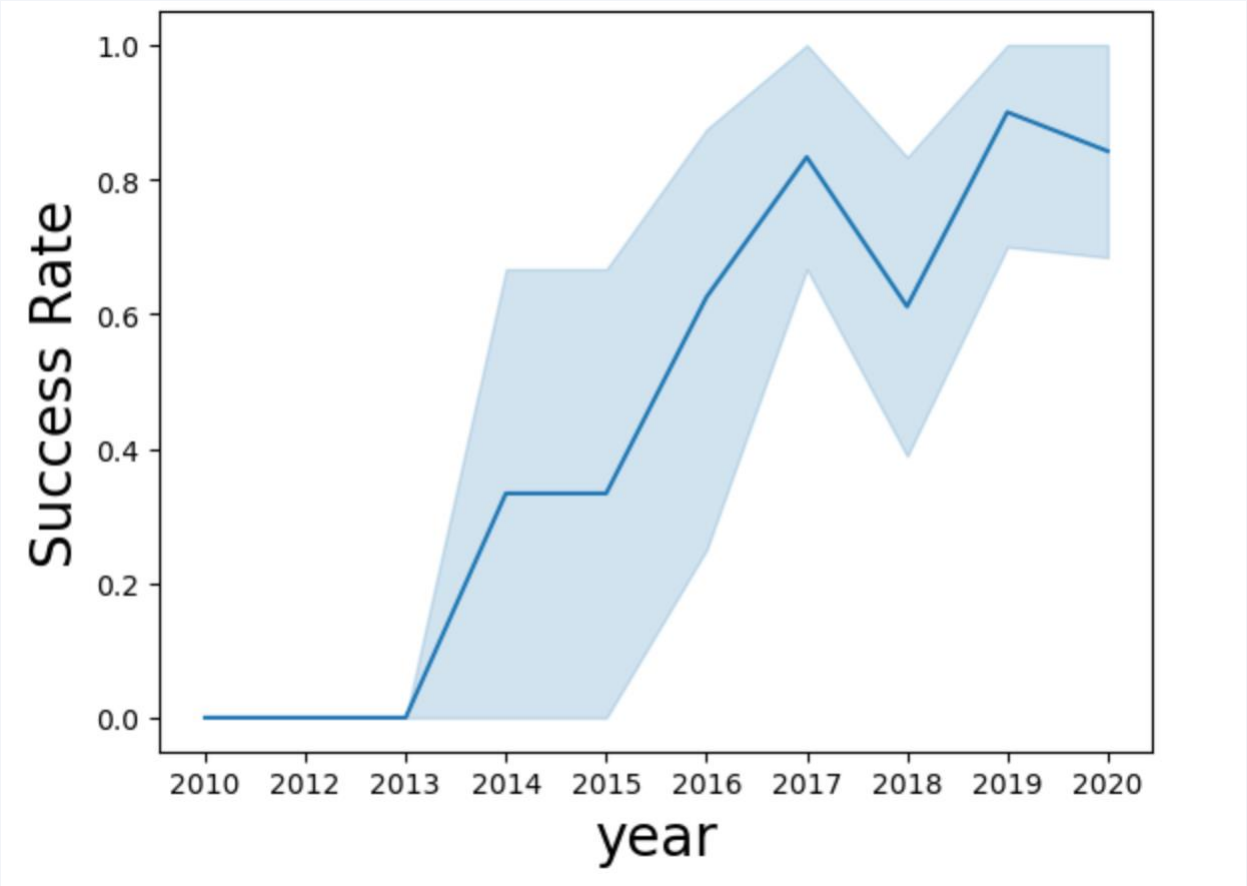
Payload vs. Orbit Type

- It is apparent that when dealing with substantial payloads, successful landings are more frequent for PO, LEO, and ISS orbits.



Launch Success Yearly Trend

- we can observe that the success rate since 2013 kept increasing till 2020



All Launch Site Names

We employed the DISTINCT keyword to display solely the unique launch sites within the SpaceX data.

Out[8]:

Launch_Site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

```
In [9]: %sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

Out[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

The query mentioned was utilized to showcase five records where the launch sites commence with the prefix 'CCA.'

Total Payload Mass

We determined that the cumulative payload transported by NASA boosters is 111268 by executing the following query.

```
In [10]: %sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD FROM SPACEXTBL WHERE PAYLOAD  LIKE '%CRS%';
 * sqlite:///my_data1.db
Done.
Out[10]:
```

TOTAL_PAYLOAD
111268

Average Payload Mass by F9 v1.1

By performing the calculation, we ascertained that the average payload mass transported by booster version F9 v1.1 equated to 2928.4 units.

```
In [11]: %sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
 * sqlite:///my_data1.db
Done.
Out[11]:
```

AVG_PAYLOAD
2928.4

First Successful Ground Landing Date

```
In [16]: %sql SELECT MIN(DATE) AS FIRST_SUCCESS_GP FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)';
 * sqlite:///my_data1.db
Done.
Out[16]: FIRST_SUCCESS_GP
         2015-12-22
```

It came to our attention that the date of the initial successful landing outcome on a ground pad was December 22, 2015.

Successful Drone Ship Landing with Payload between 4000 and 6000

We employed the WHERE clause to filter boosters that had achieved successful landings on a drone ship. Additionally, we applied the AND condition to identify successful landings with a payload mass exceeding 4000 but falling below 6000.

In [18]:
```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 AND Landing_Outcome = 'Suc
```

```
 * sqlite:///my_data1.db
Done.
```

Out[18]:

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- We observe 100 success and 1 loss

In [35]:
```
%sql SELECT MISSION_OUTCOME, COUNT(*) AS QTY FROM SPACEXTBL GROUP BY MISSION_OUTCOME ORDER BY MISSION_OUTCOME;
```

```
 * sqlite:///my_data1.db
Done.
```

Out[35]:

Mission_Outcome	QTY
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- We identified the booster that transported the highest payload by employing a subquery within the WHERE clause and the MAX() function.

```
In [36]: %sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
 * sqlite:///my_data1.db
Done.
```

Out[36]:

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

We applied a combination of the WHERE clause, LIKE, AND, and BETWEEN conditions to narrow down our search for drone ship landing failures in the year 2015. This involved identifying the respective booster versions and the launch site names associated with these occurrences.

```
In [52]: %sql SELECT substr(Date,4,2) as month, Date ,BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome FROM SPACEXTBL where Landing_Outc
 * sqlite:///my_data1.db
Done.
```

Out[52]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

We chose to focus on landing outcomes and the COUNT of these outcomes from the dataset. To refine our selection, we utilized the WHERE clause, filtering for landing outcomes falling between the dates June 4, 2010, and March 20, 2010. Subsequently, we employed the GROUP BY clause to group these landing outcomes and arranged them in descending order using the ORDER BY clause.

```
In [45]: %sql SELECT LANDING_OUTCOME, COUNT(*) AS QTY FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landir
 * sqlite:///my_data1.db
Done.
```

Out[45]:

Landing_Outcome	QTY
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Section 3

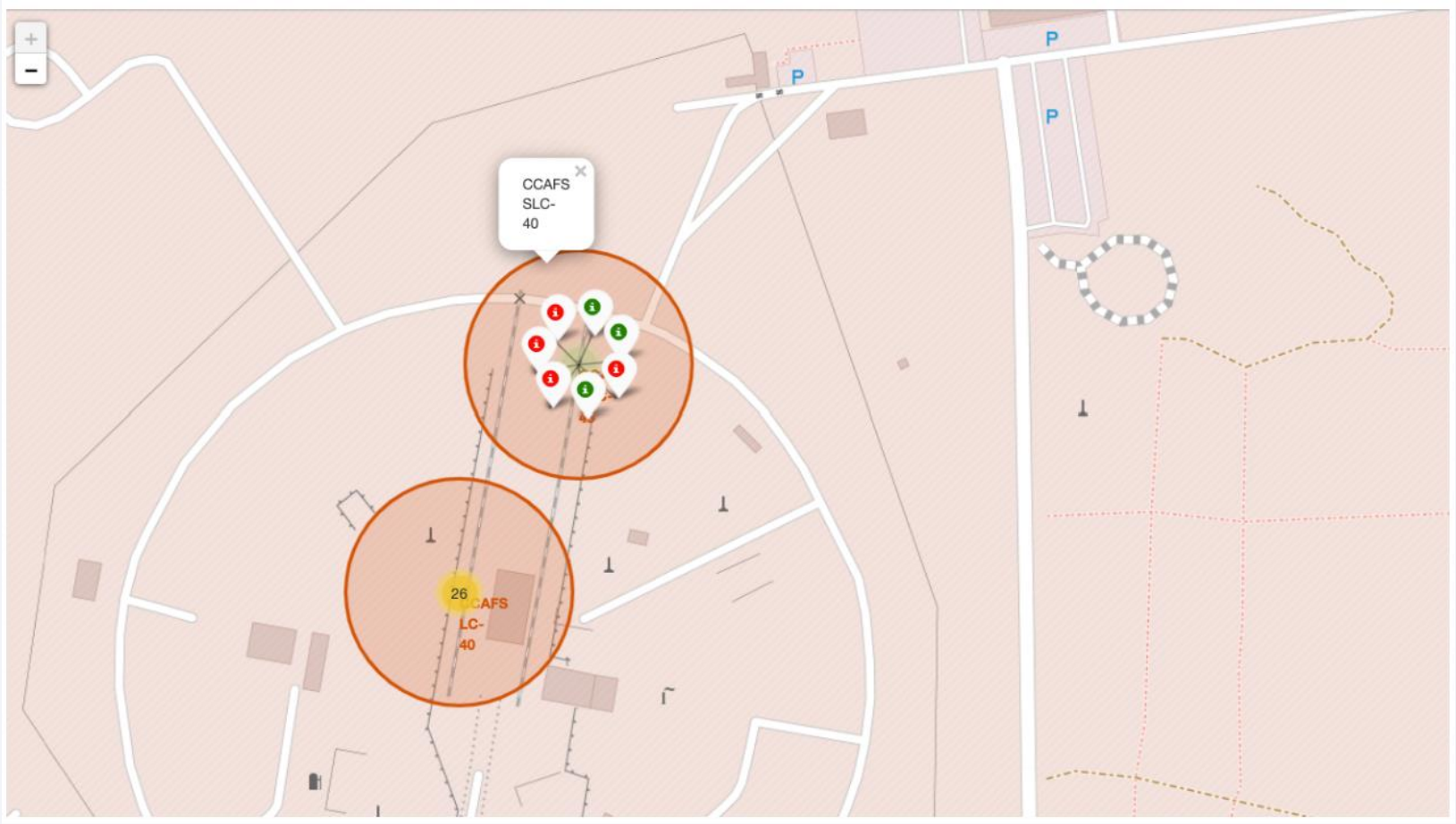
Launch Sites Proximities Analysis

Map markers for all the launch sites worldwide:

- It is evident that SpaceX launch sites are situated on the coastlines of the United States, specifically in Florida and California.

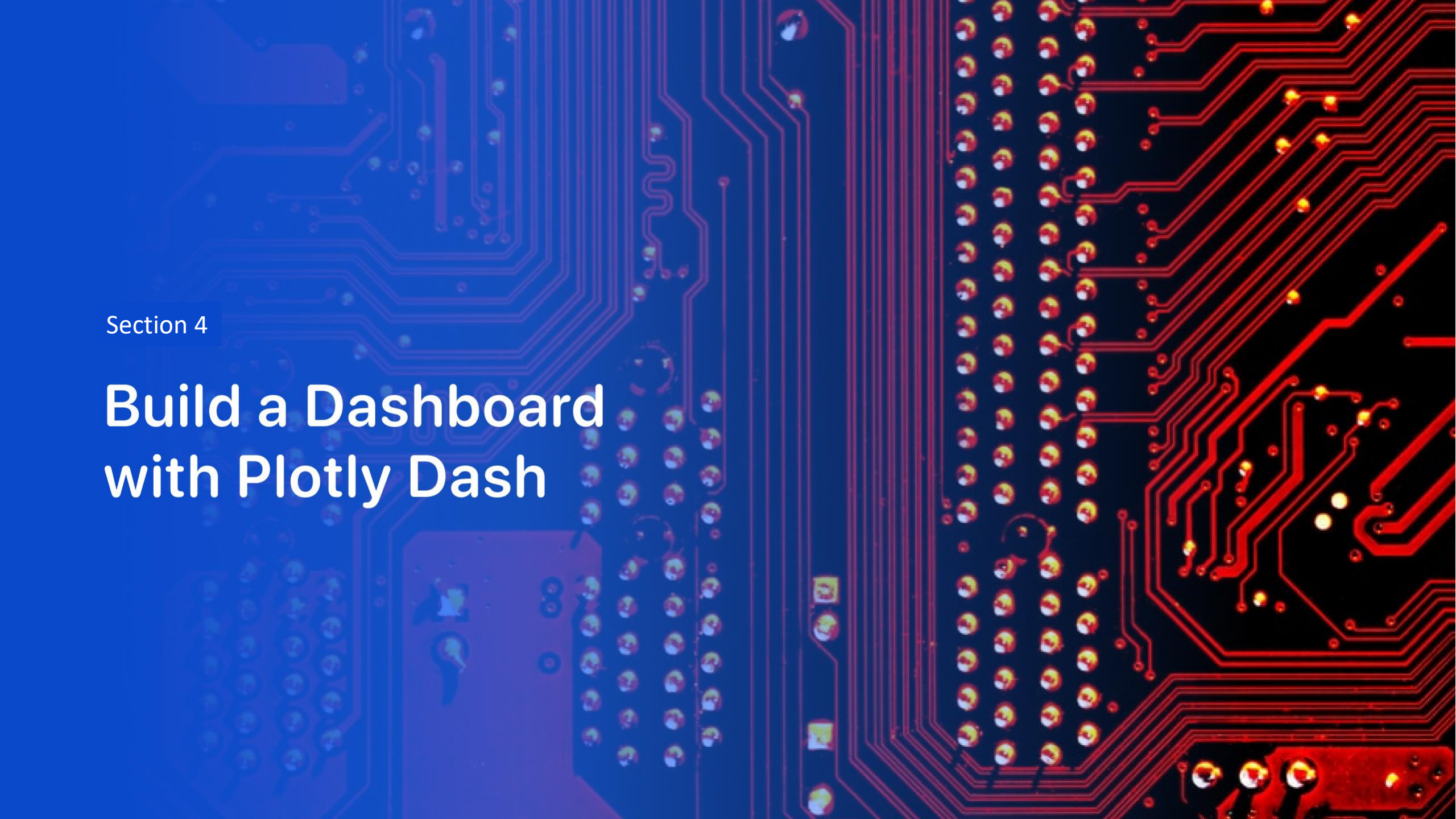
Markers that display launch sites along with color-coded labels:

Green markers indicate successful launches, while red markers signify failures.



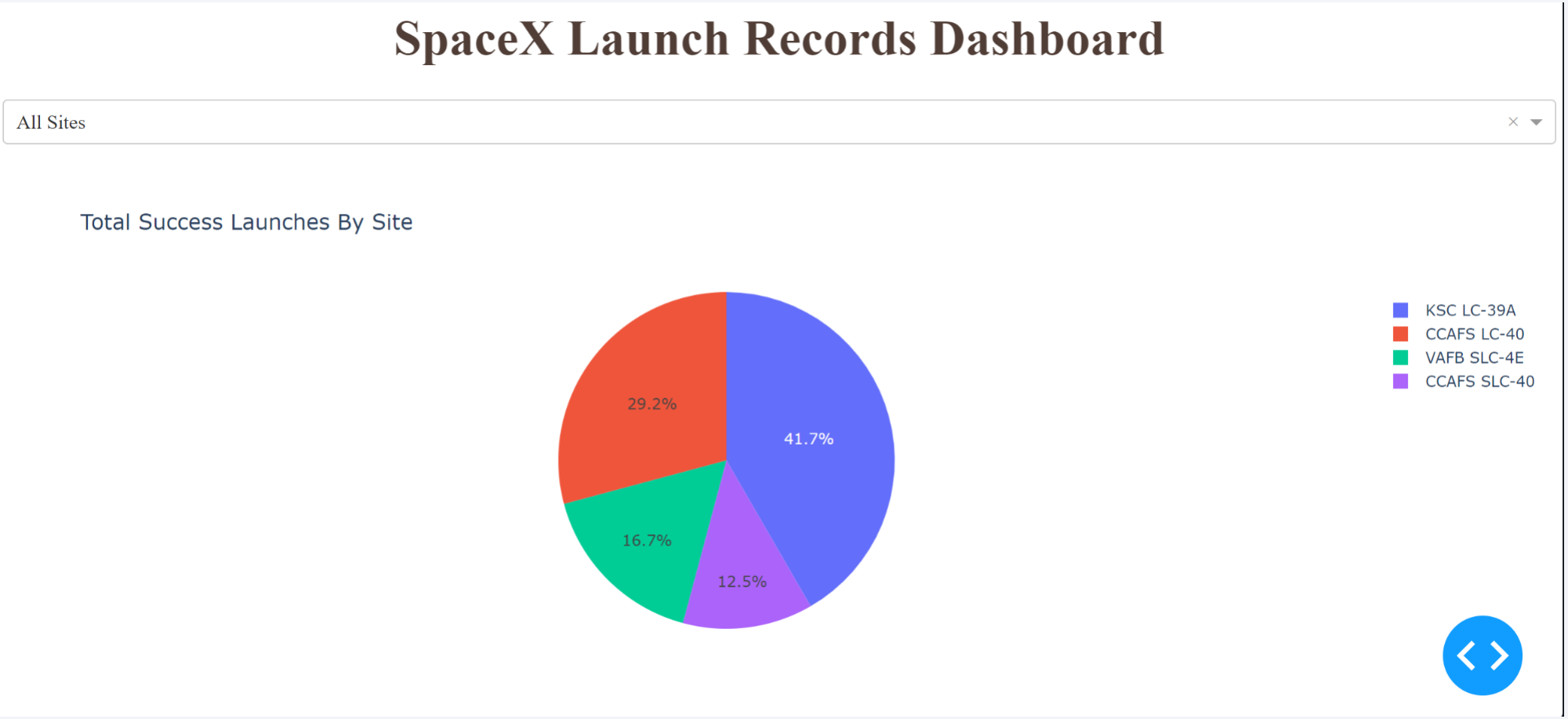
Distance from the launch site to significant landmarks

Do launch sites have nearby railway connections? No
Do launch sites have nearby highways? No
Do launch sites have coastal proximity? Yes
Do launch sites maintain a certain distance from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

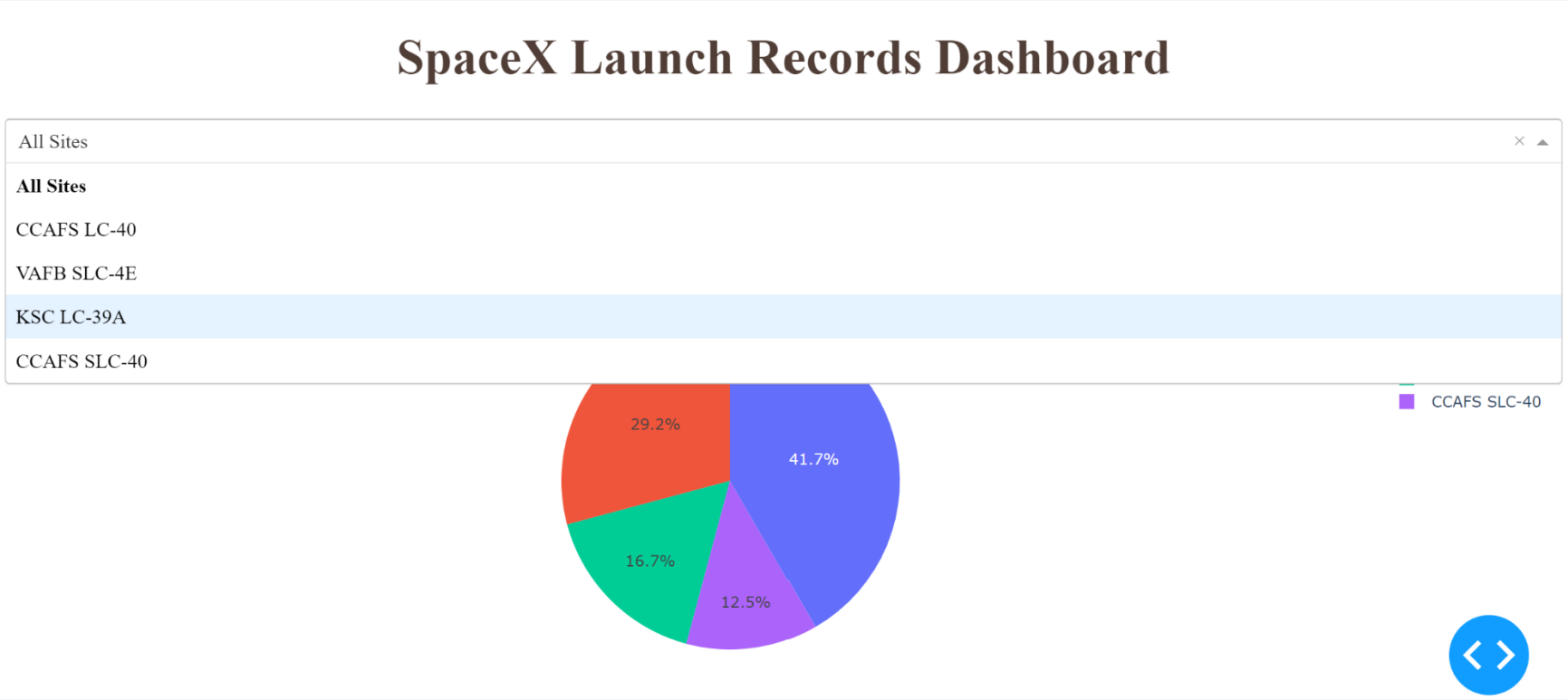
Pie chart illustrating the success rate attained by individual launch sites.

We can see that KSC LC-39A had the most successful launches from all the sites

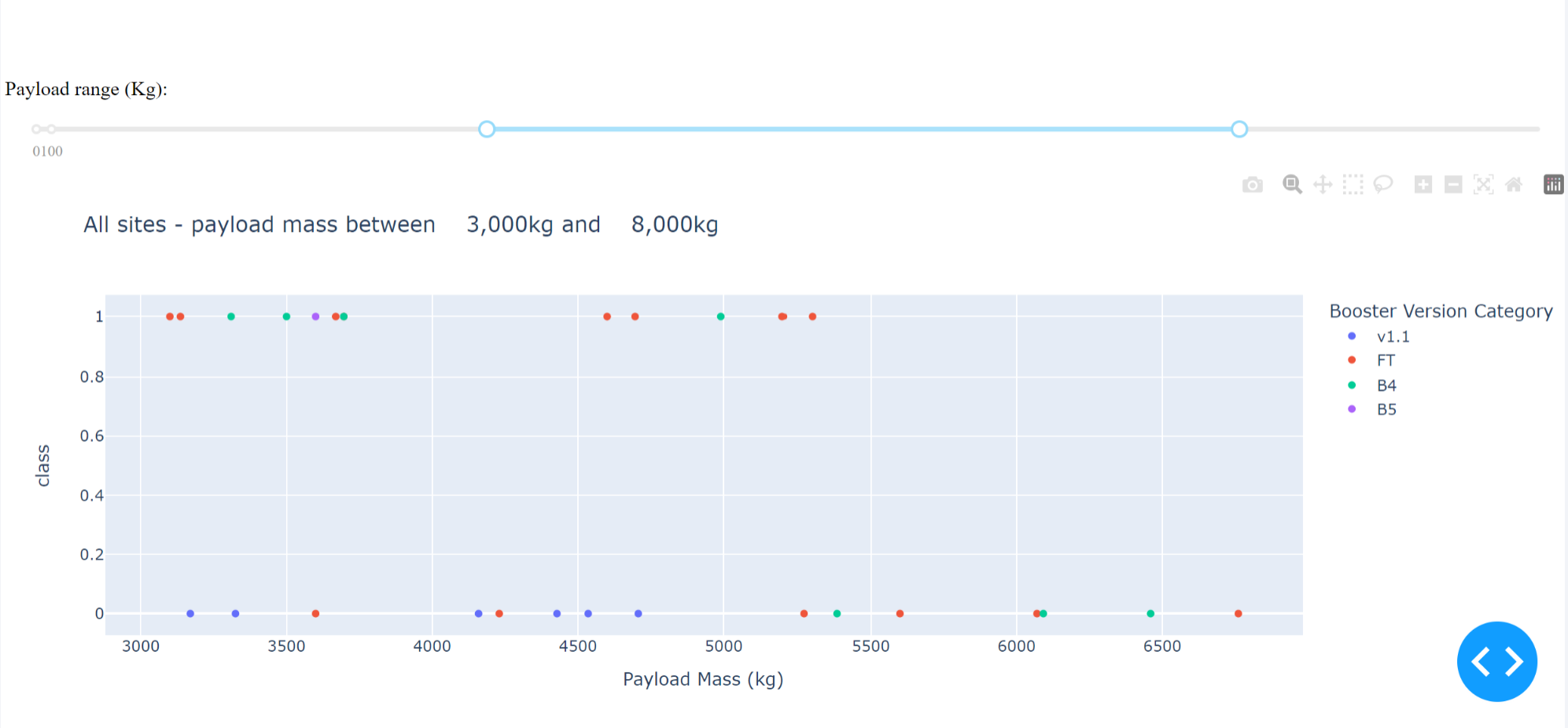


Pie chart presenting the launch site with the most notable success rate.

We visualized KSC LC-39A alone and attained a success rate of 76.9% alongside a failure rate of 23.1% .



Scatter plot depicting Payload versus Launch Outcome for all sites, where different payload values within the range slider are chosen.



Section 5

Predictive Analysis (Classification)

Classification Accuracy

The decision tree classifier is the model with the highest classification accuracy

In [ ]:

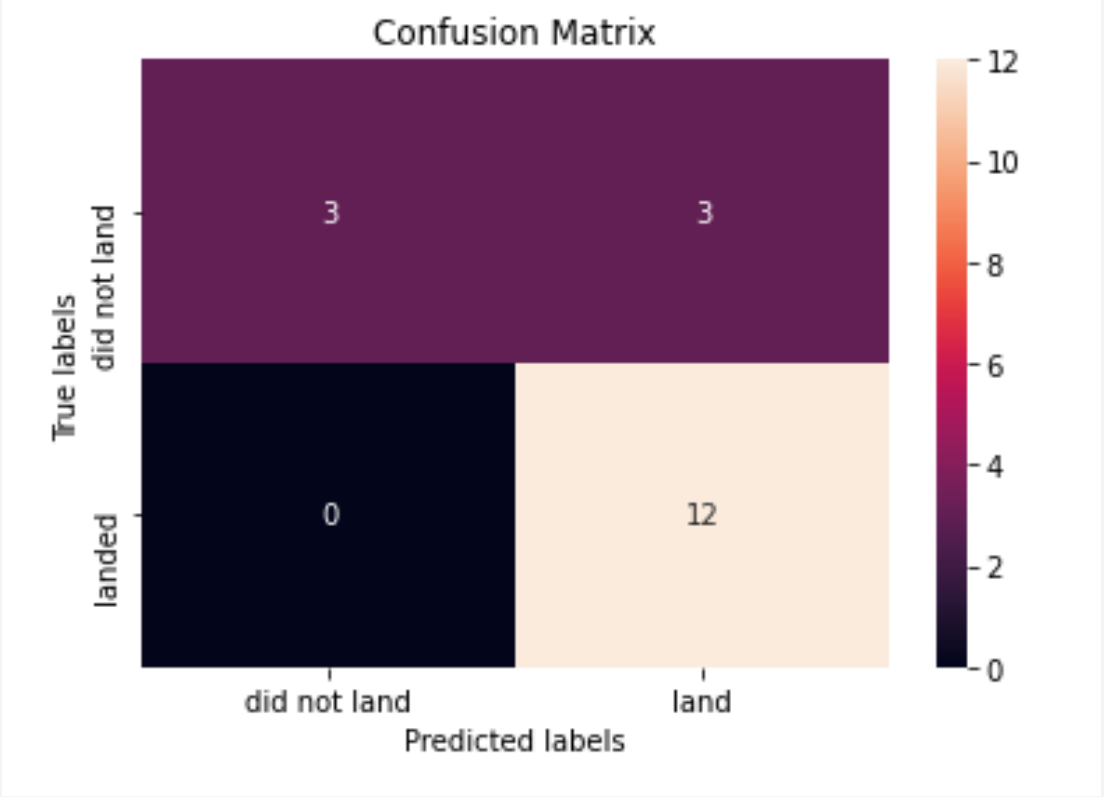
```
print("Model\t\tAccuracy\tTestAccuracy")#,logreg_cv.best_score_)
print("LogReg\t\t{}\t\t{}".format((logreg_cv.best_score_).round(5), logreg_cv.score(X_test, Y_test).round(5)))
print("SVM\t\t{}\t\t{}".format((svm_cv.best_score_).round(5), svm_cv.score(X_test, Y_test).round(5)))
print("Tree\t\t{}\t\t{}".format((tree_cv.best_score_).round(5), tree_cv.score(X_test, Y_test).round(5)))
print("KNN\t\t{}\t\t{}".format((knn_cv.best_score_).round(5), knn_cv.score(X_test, Y_test).round(5)))

comparison = {}

comparison['LogReg'] = {'Accuracy': logreg_cv.best_score_.round(5), 'TestAccuracy': logreg_cv.score(X_test, Y_test).round(5)
comparison['SVM'] = {'Accuracy': svm_cv.best_score_.round(5), 'TestAccuracy': svm_cv.score(X_test, Y_test).round(5)}
comparison['Tree'] = {'Accuracy': tree_cv.best_score_.round(5), 'TestAccuracy': tree_cv.score(X_test, Y_test).round(5)}
comparison['KNN'] = {'Accuracy': knn_cv.best_score_.round(5), 'TestAccuracy': knn_cv.score(X_test, Y_test).round(5)}
```

Confusion Matrix

The confusion matrix of the decision tree classifier reveals its ability to differentiate between various classes. However, a notable issue arises in the form of false positives, where the classifier incorrectly identifies unsuccessful landings as successful ones.



Conclusions

We can draw the following conclusions:

1. A launch site with a higher number of flights tends to exhibit a higher success rate.
2. The launch success rate has shown a consistent increase from 2013 through 2020.
3. Orbits ES-L1, GEO, HEO, SSO, and VLEO have consistently achieved the highest success rates.
4. KSC LC-39A stands out as the site with the highest number of successful launches.
5. Among the machine learning algorithms considered, the Decision Tree Classifier proves to be the most suitable for this task.

Thank you!